Avionics Architecture Standards as an Approach to Obsolescence Management.

D.J.Jibb, J.B.Walker
BAE SYSTEMS
Sensor Systems Division
Ferry Road, Edinburgh
EH5 2XS Scotland

1 Summary

Obsolescence management techniques can be categorised as either production engineering based techniques that attempt to control an existing situation or design based approaches that attempt to minimise the initial problem. This paper addresses system architecture design as an approach to obsolescence management. The work of the ASAAC programme in developing open architecture standards designed to exhibit a high level of obsolescence robustness is described. Other issues that relate to the financing and organisation of obsolescence management are also discussed.

2 The nature of the problem

Component obsolescence increasingly affects our ability to maintain military avionics in service or even to maintain production capability. The electronic component industry is now almost entirely driven by the computer, commercial telecom and consumer electronics markets. The military market is much less than 1% of the total electronics market and at this level it is no longer able to finance the high technology plants and processes that are the needed to produce specific military grade versions of state of the art commercial components. Military platform lifetimes of 30 to 40 years are, if anything, tending to increase and already are an order of magnitude higher than the typical commercial processor chip lifetime of some 2 to 3 years. The decline of the military grade component market, coupled with the rapid pace of component technology development, now requires us to find ways of using commercial quality and commercial temperature range components in military systems.

The ownership of the obsolescence problem by the whole industry and customer community will be increasingly necessary. The basic issue of obsolescence is not new. Equipment designers have been faced with the problem of making the right component choice for many years and strategies for predicting the timing of component obsolescence, and limiting its impact, are generally well developed in the avionics industry. The fundamental issue is an economic one and put very simply it is that the cost of maintaining a capability is not zero! In the past equipment suppliers have been expected to maintain a supply of components over the life of a platform. Increasingly the rate of component obsolescence is preventing this.

The Weapon System user will wish to maintain platform effectiveness in response to changing threats. Indeed it is normal for the end user to want to enhance the Weapon System through the incorporation of new or improved capabilities. Such Weapon System upgrades can provide an ideal opportunity to also manage obsolescence! In the past Weapon System upgrades were based on the mid-life update or MLU. However in today's more stringent economic climate, evolutionary upgrades based on reuse of the existing design, especially the software application designs, make more economic sense. These incremental updates spread the cost of the update programme over time and are now generally favoured. In some cases such upgrades can actually become self-financing especially if older more expensive hardware can be replaced with fewer items that conform to a newer, more capable but less expensive standard.

The funding of the obsolescence problem will require significant changes in the way we do business. In the military avionics systems of the future most of the system functionality, and in consequence the major part of the design intellectual effort, will be resident in the software. Indeed the concept of modular avionics is to standardize the hardware content, and with increasing COTS usage the hardware will come to represent relatively low value in procurement terms. However the intellectual effort needed to create the overall avionics system is likely to be greater, not less than before. To sustain the design teams needed to develop and maintain future systems it is necessary that the major deliverable, that is the software, should attract profit levels over the life-cycle which are comparable in value to those which have to date sustained the hardware based avionics industrial competence. Arguably an avionics business based solely on the supply of boxes will not be viable in the long term. Instead the industry must evolve towards capability based partnerships with emphasis on maintenance of capability on the one hand in return for maintenance of realistic margins through the supply and updating of hardware and software systems.

Producing systems with highly interactive functions and less tangible boundaries than have been customary calls for close cooperation between avionics systems suppliers and the overall system integrators or airframe manufacturers. A whole new project structure is required, in which risks can be shared and specialist knowledge pooled across a horizontal organization very different to the pyramid approach used today.

Paper presented at the RTO SCI Symposium on "Strategies to Mitigate Obsolescence in Defense Systems Using Commercial Components", held in Budapest, Hungary, 23-25 October 2000, and published in RTO MP-072.

The "integrated product team" concept, combined with team-based incentives and goals is one method of achieving the necessary critical mass of skilled and motivated design intellect.

3 Production Engineering approaches to obsolescence management

The basis of what we shall call the production engineering approach to obsolescence management is to be found in the familiar component engineering and purchasing disciplines. Specialist Component Engineers will be involved in the initial component selection process and will analyse the proposed component lists from the point of view of obsolescence so as to identify single source items or items predicted to have short commercial availabilities. For these items a stock holding and purchasing plan might be employed based on lifetime buys, or alternatively on continuous monitoring of the component availability together with last time buys as and when necessary. Increasingly the Component Engineer will be supported by access to industry component databases and in the case of a large project may exchange component availability data with other companies also involved in the programme.

Given knowledge of the exact production quantities and time-scales it should be possible to guarantee sufficient stock is held, or can be obtained, to meet the build and life-time support requirements with minimum (but not zero) financial outlay. Of course a great many circumstances can upset this ideal situation so that the stock fails to match the production build and support levels. Increased scrap rates, field failures or simply additional orders can all become problems. Again ultimately this is a financial issue since with unlimited funds sufficient stocks could be held for almost any eventuality! If all else fails the final resort is to redesign the affected assembly at the lowest level where interchangeability can be achieved so as to minimise any necessary re-qualification costs.

This paper describes a complementary approach to obsolescence management concentrating on the initial definition of the system architecture. The paper describes the work of the ASAAC programme in developing open architecture standards designed to exhibit a high level of obsolescence robustness.

4 System Architecture as an Approach to Obsolescence Management

One of the ways to break the dependence of systems on specific COTS technologies is to design systems with so called Open Architectures that provide “loose” coupling between the avionics applications and the underlying infrastructure of the computing platform. This “loose” coupling requires standardised interfaces between the application software and the hardware, system software and network interconnects so as to provide the required software portability. In addition to the software interfaces other interfaces are required to provide technology transparency in the mechanical, power distribution, network and management aspects of the system. The term System Architecture refers to a consistent set of such interfaces and the associated hardware and software building blocks. By carefully choosing the building blocks set and associated interfaces it is possible to define a stable avionics infrastructure that can potentially be maintained over the life of a platform. The design of these interfaces and building blocks is the main objective of the Allied Standard Avionics Architecture programme ASAAC. The ASAAC programme was originally set up to take benefit from the life cycle cost savings and enhanced performance potential of Integrated Modular Avionics. Given the design issues that are raised by more integrated systems, the desire to use COTS and the required long platform lifetime it is no surprise that the ASAAC architecture concepts directly result in a system architecture that is inherently more robust with respect to component obsolescence.

5 ASAAC Project

The ASAAC Phase II Programme is sponsored by the MoDs of the UK, Germany and France through a tri-lateral Memorandum of Understanding that provides for a programme to define a set of STANAGs for military core avionics. The first draft of ASAAC standards was issued in February 1999 and the current phase of work, which began in December 1999, is a 45-month programme to demonstrate and validate the standards. The remainder of this paper describes the major goals of the project and gives an overview of the top-level requirements derived from those goals. Each architecture concept area comprising software, packaging, networks and system management is described together with a short description of the relevant standards.

5.1 ASAAC Architecture Goals

The prime objective of ASAAC is to define a flexible avionics architecture that will balance affordability constraints with combat capability and combat availability. When completed, the architecture will be captured in a set of military standards (STANAGS) for IMA systems.

The three principle goals for ASAAC are,

- **Reduced Life Cycle Cost**

A major objective is to reduce the accumulated costs over the life cycle of a system i.e. the acquisition and support costs.

- **Improved Mission Performance**

The system must be capable of fulfilling the missions asked of it and satisfy all possible airborne platforms in terms of functionality, capability, accuracy, configurability and interoperability under the full scope of operating conditions.

- **Improved Operational Performance**

The goal adopted is that the system must achieve a combat capability of 150 hours (equivalent to 30 days) without maintenance with an availability of at least 95%. This goal far exceeds that achievable today and an

ASAAC system will be required to exhibit fault tolerance so that it can survive the occurrence of faults with a required level of functionality.

In addition, the maintenance philosophy dictates that modules of the system must be interchangeable between platforms of the different NATO nations and replaceable at first line.

5.2 Requirements

ASAAC has established a set of top-level requirements for an avionics architecture that are derived from the three major drivers described above. The required output from ASAAC is a set of standards for military avionics. To define those standards, ASAAC first defined a set of concepts, which described the functionality expected of a future avionics system, covering all aspects of the system from software through to packaging. From these concepts, the nature and specification of the necessary interfaces were derived. These interfaces include physical standards, software interfaces and architectural descriptions.

The top-level requirements established by ASAAC are as follows:

1. Small Set of Common Modules

The set of common modules should be reduced to a minimum to reduce development and support costs.

2. Modules Applicable to Wide Range of Platforms

The architecture should be able to support the needs of a wide variety of airborne platforms. The architecture must therefore be scaleable to be applicable to a wide range of different platforms.

3. Re-use of Software

The reusability of software between the different computational elements should be maximised.

4. Modules Replaceable at 1st Line on Aircraft

The system must be designed such that the modules can be removed at the operational site for replacement.

5. No Base and Depot Level Maintenance

The combat dependence of a platform on fixed-site airbases should be reduced or eliminated

6. Deferred Maintenance/Fault Tolerance

The architecture shall have sufficient fault tolerance to enable the system to be restored, to a predetermined level of capability, in the event of a fault at least until the next scheduled maintenance event.

7. Comprehensive BIT and Testability

The architecture shall not require tools at 1st line and shall allow on-aircraft maintenance.

8. Interoperability

Separate elements of the architecture shall inter-operate with each other.

9. Interchangeability

This requirement relates to the ability to interchange any element with any other separately developed architecture element of the same generic function.

10. Technology Transparency

The architecture should not rely on technology specific implementation techniques.

11. Use of Commercial Components, Technologies and Processes

The architecture should be designed in such a manner as to maximise the potential use of commercially available hardware and software products.

12. Maximise Digital Processing of Functions

The architecture should support the maximum amount of digital processing.

13. Functional and Physical Integration

It should be possible to attain a high level of functional integration across a physically integrated platform. This requirement aims to promote the abstraction of software applications from hardware in order to allow applications to be mapped onto various hardware architectures.

14. Open System Architecture

The architecture should exploit open commercial standards having a high-perceived level of longevity.

15. Growth Capability;

The ability to incorporate growth in technology performance and application requirements over the life of the system.

16. Modularity and Configurability;

The ability to partition a system into separate elements, each of which is individually replaceable.

In addition to these top-level requirements, a number of technical requirements were specified to ensure the usability of an ASAAC system. These requirements covered areas such as certification and qualification, security, system management and environmental requirements.

5.3 ASAAC Concept Overview

This section will provide an overview of the concepts and standards currently under definition. It is possible, and, in fact, highly likely, that the concepts will change during the programme as a result of the experience gained. All of the concepts in ASAAC are highly integrated and because of this there are numerous dependencies that make it difficult to describe the concepts clearly in a sequential manner. The software concept constitutes the most important area within ASAAC and it has therefore been described first, the other concepts following in an arbitrary order. Throughout the text there are several references to concepts that are defined in more detail later in the paper.

5.3.1 ASAAC Software Concept

The ASAAC software architecture concept defines a three-layer architecture, based on the philosophy shown in Figure 1. This philosophy describes three layers,

- **Application Layer (AL)** – representing the applications that are specific to a particular aircraft

or platform, but are independent of the enabling hardware.

- **Operating System Layer (OSL)** – representing the system software usable across all aircraft types and on all hardware architectures.
- **Module Support Layer (MSL)** – representing the hardware specific software that allows the upper software layers to be hardware independent.

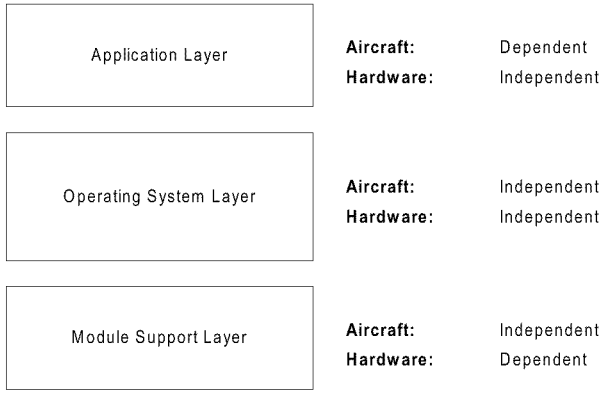

Figure 1 Software Architecture Philosophy

This philosophy requires the definition of two interfaces, as shown in **Figure 2 Software Interfaces**. These are the:

- **APOS** – the Application to Operating System interface and the
- **MOS** – the Module to Operating System interface

These interfaces provide the independence for each of the layers described previously. In addition to these layers, three functional concepts viz. Generic System Management, Blueprints and Virtual Channels were defined.

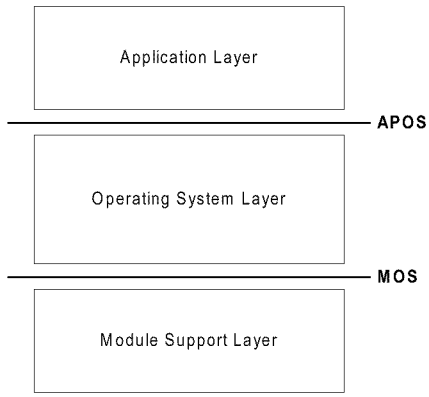

Figure 2 Software Interfaces

- **Generic System Management Concept**

The system management of an IMA system can be split into two distinct categories,

- **Application Management and**
- **Run-time System Management.**

Applications Management involves, for example, mission selection and controlling the moding aspects of a system i.e. which applications/processes need to be active during a given phase of flight. This category is implemented by the **Application Manager (AM)** located in the AL. Run-time System Management refers to controlling system initialisation and shutdown, configuration and reconfiguration, fault management, interfacing with blueprints and application scheduling. This category is implemented by the **Generic System Manager (GSM)** located in the OSL. The GSM comprises four functional elements; Health Manager, Fault Manager, Configuration Manager and Security Manager. The nature and operation of the GSM and these elements are covered in more detail in section 5.3.4.

- **Blueprint Concept**

In order to support application re-use across different hardware technologies and architectures the concept of Blueprints configuration files has been defined. Blueprints describe the mapping between the resources required by an application, in terms of processing power and communication requirements, to the available resources provided by the hardware. They are realized as a database directly accessible by the GSM. Blueprints are covered in more detail in section 5.3.1.1.

The final software architecture incorporating these concepts is shown in Figure 3. The Operating System and Extensions block includes both the operating system required for scheduling and task prioritisation purposes and other functional elements such as the Virtual Channel Manager to support communications within the system.

This software architecture requires the definition of the following interfaces,

- **SMOS** – the System Manager to Operating System interface
- **SMBP** – the System Manager to Blueprints interface and the following logical interfaces,
- **OLI** – the Operating System Logical Interface
- **MLI** – the Module Support Layer Logical Interface
- **GLI** – the GSM Logical Interface

These interfaces are described in more detail in section 5.3.1.2 Software Interface Definitions.

- **Virtual Channel Concept**

Virtual Channels (VCs) are a message-based means of communication between processes. They are designed to support the abstraction of application communication from hardware implementation. VCs are predictable in operation. In other words, an application can depend upon a VC to provide a certain set of performance characteristics, for example defined latency or bandwidth. VCs support one-to-one and one-to-many communication topologies. Other topologies, such as many-to-many, can be implemented using these two basic mechanisms. If a process on one processor needs

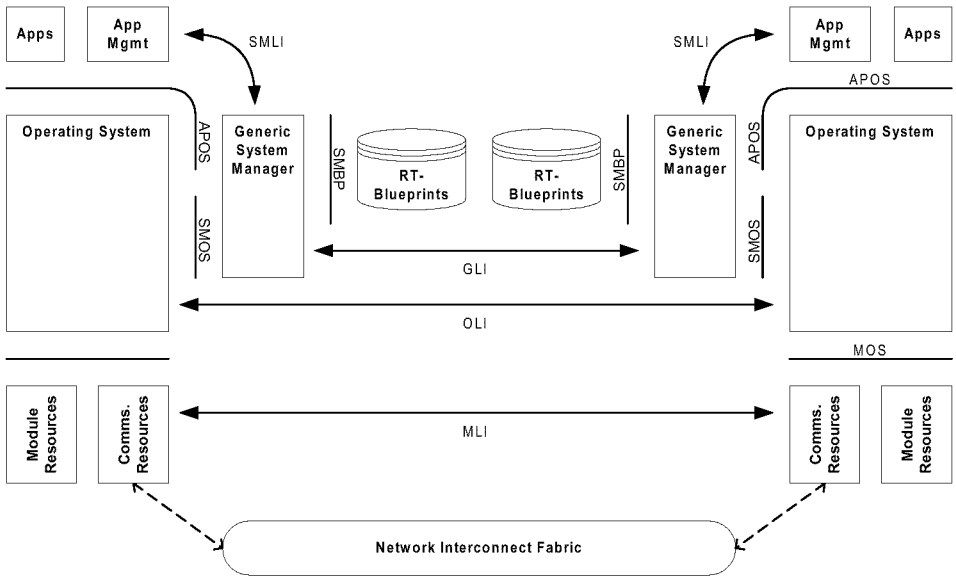

Figure 3 ASAAC Software Architecture

to communicate with a different process on the same processor, shown as type I in Figure 4 Virtual Channel Operation the communication configuration and eventual message handling are carried out by the Virtual Channel Manager (VCM) resident in the OSL. For communication between processors on the same module and on different modules, shown as types II and III respectively in Figure 4 Virtual Channel Operation, a routing service within the MSL must assist the VCM in ensuring correct transmission of the data.

5.3.1.1 Blueprints

Blueprints contain the information that defines the mapping of application processes onto the functional modules. In order to operate, the system will require a number of certified functional configurations. A functional configuration in this context refers to a mapping of application processes to module processors. These configurations will be specified at design-time and verified against the constraints placed on the system. The set of allowable configurations will be stored within the system and referenced at run time. A choice between the certified configurations will be made depending upon factors such as fault occurrence, mission status etc. In the event of a detected and localised fault or a mode change request, a reconfiguration will be performed by first selecting and then instantiating the most appropriate new functional configuration.

5.3.1.2 Software Interface Definitions

The following interfaces are defined in the software concept:

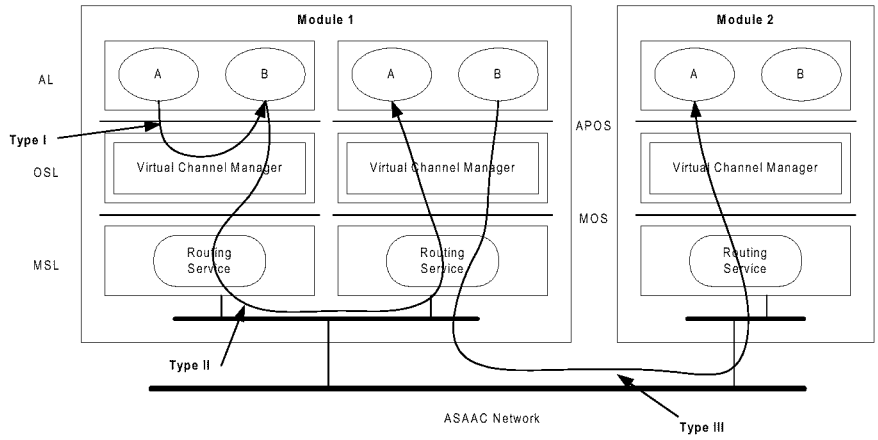

Figure 4 Virtual Channel Operation

- **APOS** – Application to Operating System

This interface is split into two sections; the Core APOS and the Specific APOS. The Core APOS applies to all module types, whereas the Specific APOS contains services specific to a particular module type.

At present, the Core APOS contains services for Virtual Channel communication, process synchronisation, timer handling, fault reporting, application management and thread management.

The Specific APOS contains services for the Graphics Processing Module (GPM), Mass Memory Module (MMM) and Power Conversion Module (PCM).

- **MOS** – Module Support Layer to Operating System

The purpose of the MOS interface is to isolate the system management software and operating system from the underlying hardware. It is envisaged that the system management software and operating system will have to be certificated and that it will be impractical to have to repeat this operation every time the hardware changes. Compliance with the MOS standard interface will allow for reuse of the System Management software and it is expected that this will minimize the need for rectification of the system management software. However it is also recognized by ASAAC that COTS OSs will exist that do not comply with the MOS. To allow these COTS products to be exploited in situations where it is not essential to preserve the integrity of the system management software ASAAC allows the use of the MOS to be optional.

- **SMOS** - System Manager to Operating System

This interface allows the GSM to access the MOS services for network configuration, process management, etc. as well as providing standard OS services.

- **SMBP** - System Manager to Blueprints interface

This interface allows the GSM to access the run-time Blueprints in order to manage configuration during system operation.

- **SMLI** - System Manager Logical Interface

This interface allows the AM to specify what application configuration is required and notify the GSM of the reconfiguration request.

- **GLI** - GSM Logical Interface

This interface specifies the message format allowing GSMs in the system management hierarchy to communicate with each other. VCs are used to transfer the messages.

- **OLI** - Operating System Logical Interface

This interface includes specification of the data presentation format to allow different operating systems to communicate with each other. Standard formats are also included for VCs and file management.

- **MLI** - MSL Logical Interface

This interface describes the network protocol and message formatting necessary for low-level communication.

5.3.1.3 Software Implementation

Although software implementation will not feature in the ASAAC standards, it is useful to give an overview of the implementation methods considered in the definition of the standards. A significant influence on the choice of software implementation in ASAAC is that of module interchangeability. This requires the ability to remove one module and replace it with another of the same generic type possibly of different implementation technology and from a different manufacturer. At present, it is expected that the common code to be executed on the modules within an ASAAC system will be stored in a central location, the Mass Memory Module (MMM). Therefore, if modules are to be interchangeable, one of the following software implementations has to be chosen,

- Produce a single binary image for every processor on every module.
- Use a Virtual Binary Interface (VBI). This is a run-time interface where executable code can expect certain functions to be resident at standardised locations in the processor memory map.
- Use an interpreted language such as Java. This would provide the ultimate in portability in that the code, at the byte-code level, would be completely reusable across any implementation of the Java (or other language) virtual machine. However, technology in this area is in its infancy and efficiency is not considered at a level suitable for avionics systems.

5.3.2 Common Functional Modules

An IMA system will consist of racks populated by Line Replaceable Modules (LRMs). One aim of ASAAC is to define a set of line replaceable Common Functional Modules (CFMs) that will be applicable to all ASAAC compliant IMA systems. Because an ASAAC module is line replaceable, it must have well defined physical and logical boundaries. ASAAC is tasked with defining the interfaces at these boundaries; in the physical sense, it is the **Module Physical Interface (MPI)**, and in the logical sense, it is the **Module Logical Interface (MLI)**.

ASAAC does not standardise on the architecture or internal interfaces within a CFM. ASAAC instead specifies two major areas of expected functionality for each module covering processing-specific and system-level functionality. The processing-specific functionality is covered by a set of requirements for each CFM type. The CFM System Support standard encompasses system-level aspects such as the system booting procedure, PBIT operation and OS download.

It is desirable for the module set to be small in size in order to maximise the potential savings in life-cycle costs. However, the task of standardising different types of processing and functionality is not simple; abstraction of the complex nature of modern technologies and dealing with differences in architectures and designs is extremely problematic.

In ASAAC, six different CFM types have been defined,

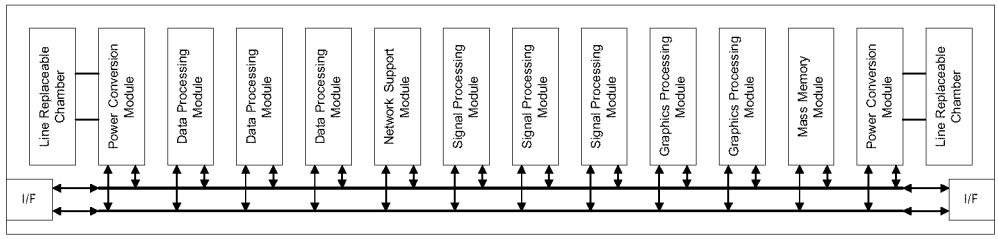

Figure 5 ASAAC Rack Arrangement

- **DPM – Data Processing Module**

The DPM covers the data-dependent data processing activities expected of the IMA system.

- **SPM – Signal Processing Module**

The SPM covers the data-independent processing activities of the IMA system, such as DSP based front-end signal processing.

- **GPM – Graphics Processing Module**

The GPM provides symbol-based graphics generation and image composition and formatting.

- **MMM – Mass Memory Module**

ASAAC promotes the use of a central facility for program storage for portable code. In addition, IMA systems have a large non-volatile storage requirement for capabilities such as terrain information, EW information, mission planning etc.

- **NSM – Network Support Module**

The NSM provides the network upgradeability, in terms of protocol and/or network control, by locating the active components for the network within a line replaceable module.

- **PCM – Power Conversion Module**

The PCM acts as the first stage in a two-stage power conversion architecture converting raw aircraft power to 48V for distribution across an avionics rack backplane.

An ASAAC rack (see Fig 5) is expected to comprise:

- **A Single NSM,** to provide the network routing,
- **A Single MMM**, to provide initialisation control and program download,
- **Multiple DPMs**, as the general processing resource,
- **Multiple SPMs**, as the signal processing resource,
- **One or two GPMs**, as the graphics processing resource, and
- **Two PCMs,** to provide dual redundancy on the power inputs.

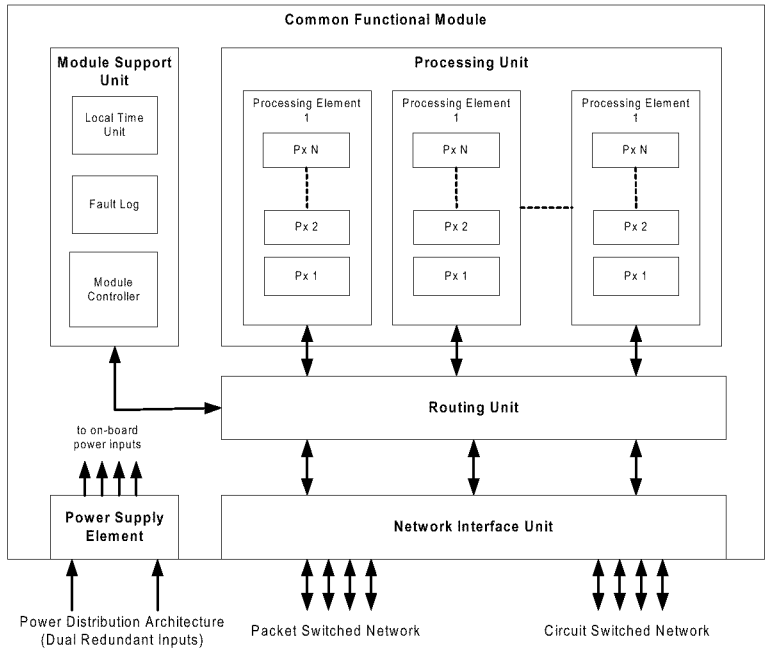

Figure 6 Generic CFM Architecture

5.3.2.1 Generic CFM Concept

The DPM, SPM, GPM and MMM module types outlined above adhere to the Generic CFM Concept shown graphically in Figure 6 Generic CFM Architecture. This concept was devised to promote re-use of hardware and software elements for module manufacturers. The Generic CFM Concept defines the following functional units:

- **MSU – Module Support Unit.**

The MSU is responsible for supporting certain system activities, specifically System Initialisation and Fault Management. The MSU also supports generic functionality required of each ASAAC-visible processor, i.e. one that executes the ASAAC software stack, such as time synchronisation and fault logging. The MSU contains a programmable resource, termed a Module Controller. Together with non-volatile memory used for status, BIT and fault logging. The MSU is also responsible for standard time distribution, which is covered in more detail in section 5.3.5.

- **NIU – Network Interface Unit**

Each module will interface to the standard ASAAC network through the MPI and MLI. Although each module type will likely have different network interface requirements, in terms of number of links and link capacity, a significant amount of the network interface should be common between module types assuming the same network. The NIU is responsible for acting as the primary network interface on the module and converts the on-board communications to the format required by the ASAAC network. The present network concept in ASAAC refers to a Packet-Switched and a Circuit-Switched network; these are covered further in section 5.3.3.

- **RU – Routing Unit**

The RU represents the internal communication within a CFM. The Routing Unit implies no architecture or implementation; it only describes the functional capability that allows all the other units to communicate with each other.

- **PU – Processing Unit**

The PU represents the processing specific functionality for each of the module types; data, signal, graphics processing or mass memory.

- **PSE – Power Supply Element**

48V is provided, as standard, to each module from the backplane. It is the responsibility of the PSE to convert this input to the voltage levels required on the module. Each CFM will have dual-redundant power inputs and the PSE shall be able to consolidate these.

5.3.3 Network

The networking requirements for IMA systems differ greatly from those of previous federated systems. The splitting of processing functions that were previously located within single units and the trend to higher digitisation rates give rise to a larger required total network bandwidth. In addition, the requirements that the system support fault tolerance demands that the network support a high level of mobility of software

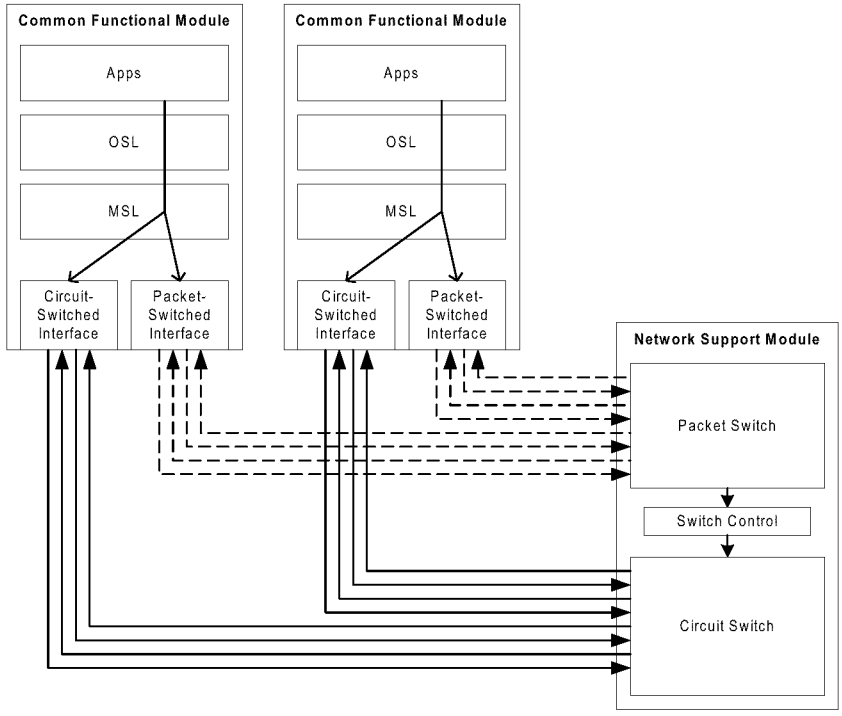

Figure 7 ASAAC Network Architecture

application functions within the physical system. The top-level requirement for module interchangeability implies that the communication network must provide standardised interfaces and operations. At the same time the network concept must support technology transparency and provide for the incorporation of COTS technology with potential for scalability and growth. A unified approach to the network for the whole IMA system is desirable in order to avoid the proliferation of hardware and software elements for different network types.

Parallel electrical bus technology is perceived to now be at an upper limit as far as the module interface is concerned. Serial protocols are very much preferred because of their routing and growth capabilities and because of the predicted transition from electrical to optical transmission media. Although optical media for module interconnection has still to be proven for extensive use in an avionics environment, especially the choice between single- and multi-mode technologies, it is almost inevitable that it will be fully adopted for IMA systems in the future. ASAAC has made the use of optical media mandatory for the standards demonstration and validation to be performed as part of the project.

The current baseline for the network describes two distinct network components viz.

- **Circuit-switched network** and
- **Packet-switched network**

These two networks make use of an NSM to provide the routing and link reconfiguration.

The circuit-switched network is implemented using unidirectional SDH STM-16 point-to-point links. This network is aimed for high-bandwidth data-streaming applications. The particular version of SDH used, STM-16, should provide a single link bandwidth of 2.488 Gbit/s. The NSM will contain protocol-independent switches to provide the routing.

The packet-switched network will be implemented using ATM on top of an SDH STM-4 physical layer. This network is aimed at the lower bandwidth applications requiring high routing flexibility. SDH STM-4 can be expected to provide a bandwidth of 622 Mbit/s. The NSM will contain the ATM switches necessary to provide the routing.

The interconnection architecture between modules using the NSM is shown in Figure 7 ASAAC Network Architecture Each module communicates via packet-switched and circuit-switched network interfaces. Any communication from an application is directed to the relevant interface by the MSL on the processor.

The standards relevant to the network are the

- **MPI** – the module physical interface to the network, and the
- **MLI** – the module logical interface to the network.

5.3.4 System Management

System Management builds upon the other concepts to allow an IMA system to operate. In ASAAC, the System Management concept does not relate directly to a standard however elements of the concept are implemented in other standards, such as the SMOS and CFM System Support. The majority of the concept is defined in the System Management Guidelines. In essence, the concept exists as a recommended method of implementation describing sets of functionality rather than as a set of interface standards.

System management must be able to control the configuration and operation of an avionics system at processor level, and at the higher rack or integration area levels. To achieve this, the system management concept in ASAAC follows a hierarchical approach, maximising the modularity and re-use of functional elements. The following levels of hierarchy are defined,

- Aircraft level
- Integration Area level
- Resource Element level

Bi-directional communication exists between the levels of system management to provide control and reporting paths that allow a coherent view of the system. The system management concept is implemented by the Generic System Manager (GSM) and Application Manager (AM) in the software concept. The GSM is resident within the OSL of the software stack and the AM within the application layer. The hierarchy is shown in Figure 8 System Management Hierarchy.

.

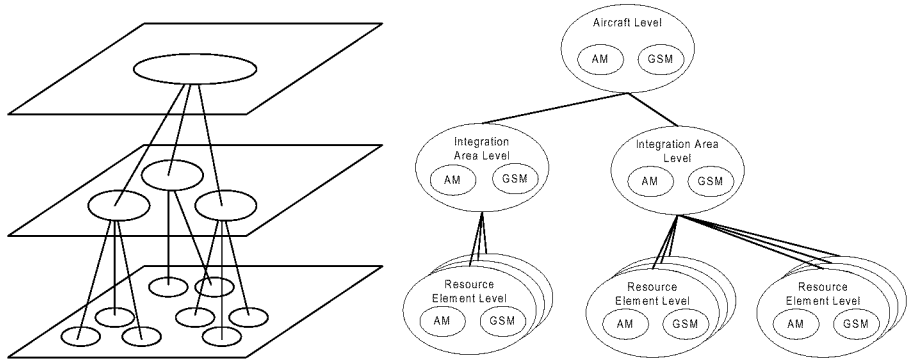

Figure 8 System Management Hierarchy

The GSM will contain the following functional elements,

- **Health Monitor (HM)** – This element is responsible for monitoring the health of the system. It receives input from the Fault Manager located in the next level down in the management hierarchy, as shown in Figure 9 GSM Interactions. After processing of the fault reports, it will indicate to the Fault Manager located in the same level of hierarchy if the HM believes a persistent fault to exist.
- **Fault Manager (FM)** – This element is responsible for collating fault reports from the HM and reporting to the HM in the management layer above. The FM then determines the action to be taken and makes a request for reconfiguration to the Configuration Manager. The nature of the request is dependent upon the nature of the fault report.
- **Configuration Manager (CM)** – This element is responsible for co-ordinating the requested reconfigurations from the FM. The CM in one level communicates with the CM in the lower level to manage the reconfiguration.
- **Security Manager (SM)** – This element performs fairly independently of the other elements and is responsible for control of access rights for input and output requests within the relevant management area, be it aircraft, integration area or resource element. The nature of security in an IMA system is covered later in this section.

A resource element is conceptually defined as the lowest level of the management hierarchy. In an implemented system, it is expected that each processor hosting an ASAAC software stack will be defined as a Resource Element.

For aircraft and integration area managers, a GSM on a processor or resource element within that area will be nominated as the area manager. This nomination occurs at system initialisation or if necessary after a reconfiguration resulting from a failure.

5.3.5 System Time

In order to synchronise the management tasks and applications within the system, it is necessary to maintain an absolute time clock that is available to all elements within the system. To achieve this, a Master Reference Clock (MRC) is used to distribute a time signal to Reference Clocks located on the modules.

5.3.6 Reconfiguration

The reconfiguration concept refers to the following definitions,

- **Configuration** – a static state of the system, with certain processes executing on certain processors.
- **Reconfiguration** – the set of actions that need to be executed to perform a transition from one configuration to another configuration.

There are two distinct situations where a reconfiguration will be initiated,

In the event of a mode change request – here the task is simple; the System State is known so the reconfiguration process to the required new configuration is known.

In the event of a fault – here the situation is more complex. If a fault has occurred, then the system is in an unknown state. This state must be analysed and verified before reconfiguration to a known configuration can be carried out. The present concept for reconfiguration is that all the possible (and relevant) configurations of processes on processors are stored in the blueprints. Upon initialisation or reconfiguration, the most suitable configuration is chosen from the Blueprints.

5.3.7 System Initialisation

The initialisation of the system occurs in three stages. First, a generic procedure is executed to provide a limited initial capability. Second a mission dependent

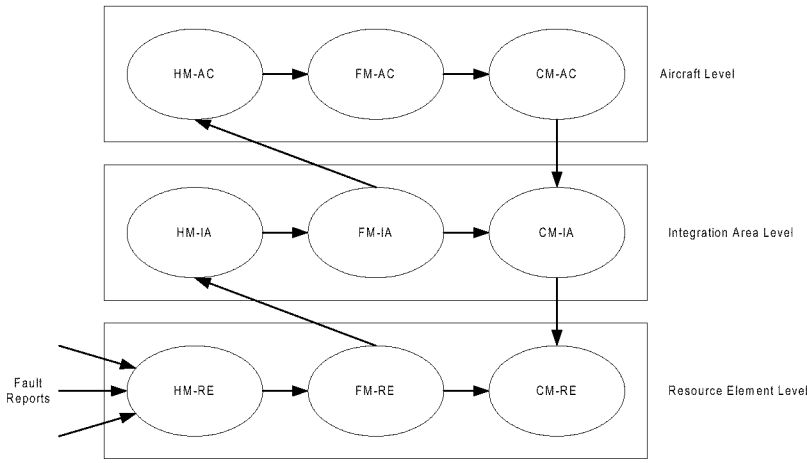

Figure 9 GSM Interactions

initialisation configures the system into partial operational mode to allow activities such as refueling and maintenance. Finally, a detailed mission-oriented initialisation configures the applications to provide the full avionics functions. These initialisations are essentially a sequence of reconfigurations that build up levels of functionality at each step. The present concept is to utilise a MMM as a 'bootstrap' module to initiate this process in collaboration with a DPM and an NSM.

5.3.8 Fault Management

Fault Management is the methodology of handling faults within a system in order to prevent total or partial system failure. It encompasses the following aspects,

- **Fault Tolerance (FT),** which allows continued operation of the system in the presence of faults
- **Integrated Test and Maintenance (ITM),** which allows identification of the failed component for repair.

FT has the further responsibility to ensure survival of the system with a suitable level of functionality in the event of a fault.

5.3.9 Security

In general, security is concerned with the protection of assets from threats where a threat is defined as the potential for abuse of the protected assets. Because of the highly integrated nature of an IMA system, functional areas with very strict security requirements such as communications and navigation are brought into close contact with other areas such as radar and vehicle management. IMA systems must, therefore, be able to deal with differing security requirements across common equipment and ensure sufficient asset protection. There are two major areas of security,

- **Communications Security (COMSEC)** – for COMSEC, a functional interface to a cryptographic functionality will be defined
- **Computer Security (COMPUSEC)** – for COMPUSEC, the functionality can be located either wholly in software or spread between hardware and software.

It is highly likely that compliance with the Common Criteria (CC) for Information Technology Security Evaluation will form part of the accreditation for an IMA system.

5.3.10 Safety and Certification

The key characteristics that an ASAAC system must demonstrate to support safety certification are data integrity, guaranteed availability of data and resources, and predictability of operation. The higher integration that is achieved through the use of IMA will mean that safety/certification issues, similar to security issues, will begin to affect a larger number of functional areas.

At this stage of the ASAAC project, the issues regarding security and safety and certification are not fully defined. Task forces are at present continuing to investigate the possible consequences that the various requirements will have on an IMA system and have yet to produce an agreed approach for recommendation.

5.3.11 Packaging

The physical outline and connector configuration of a module could be considered one of the most important aspects of an IMA system. For modules to be interchangeable at all there must be a standard physical interface. The ASAAC packaging concept defines the Module Physical Interface (MPI), which covers the packaging, cooling, power supply distribution, electromagnetic compatibility and interconnection standards.

5.3.11.1 Module Packaging

The ASAAC packaging baseline standard specifies a module format similar to Double Eurocard, termed ASAAC A. However, ASAAC A specifies a short-side connector as opposed to the long side for traditional VME. Where compatibility could be considered is in the usable area of a module. In that case, COTS board designs could be ported to the standard physical outline with greater likelihood of success. Because of the concept defined for the CFMs in ASAAC, there will be a minimum area suitable for providing the defined functionality. Also, the fact that CFMs are to be line replaceable implies that they should be of a certain manageable size. It is generally felt that Double Eurocard is approximately the correct area in which to provide a CFM with a significant capability.

5.3.11.2 Module Cooling

Two cooling techniques have been chosen for the present baseline,

- **Conduction cooling**, with the possibility to use heat pipes in the module core to enhance the performance.
- **Airflow cooling**, with air circulating either outside the CFM (air flow around), through a central CFM heat exchanger (air flow through), or with air directly on the module components (direct air flow).

These options are chosen to provide the widest range of alternatives between affordability and performance, thus catering for anticipated module power dissipation levels and different platform cooling systems.

- **Module Interconnect**

ASAAC only allows optical interconnections external to a module, except for power distribution. For optical interconnect, two technologies have been considered,

- **Embedded Fibre** – the optical fibre is placed onto an adhesive coated substrate and an additional protective layer mounted on top. Complex topologies are possible, including star couplers.
- **Polymer** – this technology is the fabrication of flexible flat sheets of polymer containing optical waveguides, which can be very cost-efficient if produced in large quantities.

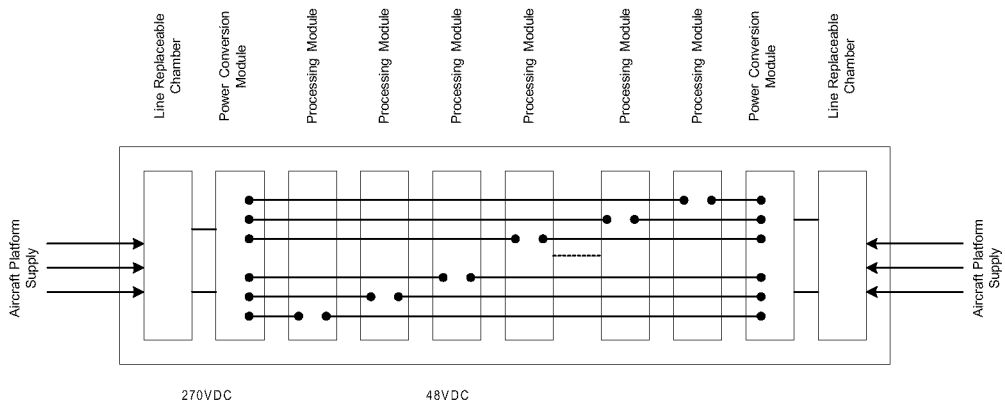

Figure 10 Power Supply Architecture

- **Module Connector**

ASAAC will define a connector shell capable of accommodating either butt coupled or free-space inserts. The connector shell consists of aluminium support shells, each shell having three main cavities. At each end of the module shell a polarised guide pin is positioned with the corresponding guide socket on the backplane shell. Each shell cavity can accommodate a variety of inserts; standard size 22 signal contacts, high density PCB signal contacts, size 16 power contacts and 32 or 48 fibre-optic contacts, depending on density. Therefore, a customisable connector can be manufactured that is tailored to specific system requirements.

5.3.12 Power Distribution Architecture

The power distribution architecture within ASAAC is a two-stage conversion process, with conversion from the aircraft platform supply to an intermediate internal rack voltage level, and subsequent conversion to logic voltage level at each module. A Line Replaceable Chamber (LRC) converts the aircraft platform supply to 270VDC and performs the supply filtering. The Power Conversion Module (PCM) converts the 270VDC supply to a rack standard of 48VDC. This scheme is shown in Figure 10 Power Supply Architecture.

48VDC was chosen because it is a common commercial standard and possesses inherent safety and support for hot plugging and unplugging of modules. Note that, in the architecture, each processing module has dual redundant supplies to enhance fault tolerance. Each processing module will perform on-board supply consolidation and conversion to appropriate logic levels.

The PCM will possess load current-monitoring capabilities in order to detect faulty power circuits on modules.

5.4 Standards Under Definition

The standards under definition by ASAAC are listed in Table 1.

Standard Name	Status
APOS	First draft available
MOS	First draft available
SMOS	First draft available
SMBP	First draft available
GLI	First draft available
OLI	First draft available
MLI	First draft available
CFM System Support	First draft available
MPI	First draft available

Table 1 List of ASAAC Standards

6 Conclusions

This paper has described the ASAAC Avionics Architecture that is being developed to provide the technical basis for advanced avionics for new platforms and updates from around 2003 onward. A carefully designed System Architecture can provide a stable structure within which COTS components and processes can be accommodated with reduced risk from obsolescence. The interfaces reduce the coupling between the application software, which is the major repository of value in the avionics system and the underlying hardware, software and network components. A system designed around IMA concepts will be much easier to upgrade and consequently more resilient to component obsolescence. Maintaining the capability of an avionics system in the future will entail regular expenditure on technology insertion activities that will provide benefits in terms of performance and at the same time will contribute to the management of component obsolescence.